RUP 기반의 웹 마이그래이션 설계 모델 재사용에 관한 연구

민사역°, 조현훈, 류성열
숭실대학교 컴퓨터학과

{minsy, hhcho}@selab.soongsil.ac.kr, syrhew@computing.soongsil.ac.kr

A Study on the Reuse of Design Model for Web Migration based on RUP

Sa Yeok Min°, Hyun Hoon Cho, Sung Yul Rhew
Dept. of Computing, SoongSil University

요 약

최근 인터넷은 사용자에게 각광을 받고 있다. 이유는 인터넷이 되는 곳이면 어디서든 웹 브라우저만 있으면 다른 프로그램 없이도 접근이 가능하고 모든 서비스를 제공 받을 수 있다. 기존의 어플리케이션(C/S 어플리케이션)은 서버와 클라이언트를 따로 설치해야만 하는 번거로움이 있었다. 또한 프로그램의 변경 시 서버나 클라이언트 어플리케이션의 재 설치가 필수적이다.[1] 따라서 최근 업계에서는 인터넷을 통한 서비스를 위해 많은 컨텐츠 개발이 이루어지고 있다. 이런 흐름은 기업의 그룹웨어나 어플리케이션 등을 인터넷을 통해 서비스하도록 만들고 있으며, 기존의 어플리케이션과는 관계 없이, 즉 거의 모든 것을 재사용하지 못하고 처음부터 다시 개발하고 있는 실정이며, 사내에서는 기존 어플리케이션을 그대로 사용하면서 하나의 웹 서버를 이용하여 외부의 클라이언트에게 서비스를 해 주고 있는 실정이다.

본 논문에서는 기존의 어플리케이션 산출물, 즉 분석, 설계 문서들을 가지고 웹 마이그래이션 시 재사용할 수 있는 범위와 산출물을 추출하고 소프트웨어의 아키텍쳐를 재정의하고, 웹 마이그래이션을 위한 설계 모델에서의 재사용할 수 있는 프로세스를 제시하고 각 프로세스에 대한 활동과 지침을 정의한다. 또한, 사례 연구를 통해 적용해 본다.

1. 서론

기존에 개발된 C/S 프로그램을 웹에서도 서비스를 가능하게 하기 위한 절차와 지침을 마련함으로써 더 효율적으로 웹 마이그래이션을 성공할 수 있다. 따라서 본 논문에서는 기존 시스템에 대한 분석, 설계한 산출물을 가지고 재사용할 수 있는 부분을 추출하는 절차(Process)를 제시한다. 또한 이 절차에 의해 나오는 산출물을 실제 구현에 적용하므로 코드 수준에서도 어떻게 적용이 되는지 확인할 수 있으며, C/S 어플리케이션의 분석, 설계 산출물은 웹으로 마이그래이션 한 어플리케이션에도 일관성 있게 재사용이 된다.[1]

본 논문이 제시하는 기존 시스템의 재사용 절차에서는 분석, 설계의 결과로 나오는 산출물인 유스케이스 다이어그램(Usecase Diagram), 시퀀스 다이어그램(Sequence Diagram), 클래스 다이어그램(Class Diagram) 등을 사용자 인터페이스(User Interface), 입출력(Input/Output), 클라이언트 비즈니스 로직(Client Business Logic), 서버 비즈니스 로직(Server Business Logic) 등으로 나눠 분석함으로써 웹 어플리케이션으로 변환 시 반드시 고려되어야 하는 사항을 정의하고 추출하여 해당되는 부분만을 재 설계하고 코드 수준에 반영함으로써 산출물과 코드에 대한 재사용의 효율을 증가시킬 수 있다. 또한 이러한 일련의 과정은 개발 비용의 절감을 가져온다.[2]

2. 관련 연구

2.1. RUP(Rational Unified Process)

미국의 Rational 사에서 제안한 소프트웨어 개발 프로세스로 소프트웨어의 품질을 효과적인 개발을 위한 지침을 제공한다.

RUP 의 구성은 6 개의 핵심 프로세스 워크플로우와 4 개의 단계로 구성되며, 이 단계는 6 개의 핵심 프로세스 워크플로우에 점진적이고 반복적으로 수행된다.[3] 이와 같이 구성된 RUP 는 기존 시스템인 C/S 어플리케이션을 웹 마이그래이션 하는 모든 단계에 적용되며, 세부적인 워크플로우와 활동(Activity) 및 산출물(Artifact)에 대해서는 변형을 통해 적용되었다.

2.2. 4+1 View Model of Architecture

소프트웨어 아키텍쳐 중심의 프로세스로써, 개발하고자 하는 시스템이나 기존에 개발된 시스템을 이해관계자(stakeholder) 각각의 관점으로 시스템을 이해하기 쉽게 모델링하는 프로세스이다.[3,4] 아래는 각 관점에 따른 아키텍쳐의 분류이다.

- Logical View : 분석가/설계자

- Implementation View : 프로그래머
- Process View : 시스템 통합자
- Deployment View : 시스템 기술자
- Use-Case View : 사용자

2.3. WAE(Web Application Extension)

최근 수 많은 웹 관련 기술이 발표되고 웹 어플리케이션이 더더욱 복잡해지고 미션 크리티컬한 특성을 가지므로 웹 어플리케이션 개발을 위한 모델링이 필요하게 되었다. 또한 웹 어플리케이션의 복잡성이 더해지므로 기존의 간단히 개발되던 웹 어플리케이션이 여러 명의 팀 프로젝트로 발전함에 따라 효율적인 개발 프로세스를 지원하는 모델링 필요성이 대두되었다.[4,8] 이러한 문제를 해결하기 위해 Jim Collanen 은 UML의 스테레오 타입을 이용한 웹 관련 기술 및 웹 페이지에 관한 모델링 방법을 제시했다.[5,6,7]

2.4. 관련 기술

2.4.1. RMI(Remote Method Invocation)

RMI 는 소켓을 사용하지 않고 분산 컴퓨팅 환경에서 네이밍 서비스(Naming Service)를 통해 서버의 서비스 원격 객체를 얻어옴으로써 분산된 클라이언트가 서비스를 받을 수 있는 전형적인 분산 클라이언트/서버 프로그램 기술이다.

2.4.2. Servlet

서블릿은 애플릿의 반대 개념의 웹 기술로써 웹 서버에서 실행이 되며, 클라이언트 요청에 하나의 프로세스를 생성하는 것이 아니라, 쓰레드를 생성함으로 작업 부하면이나, 속도면에서 성능을 향상시킨다. 또한 서블릿은 클라이언트의 요청을 받아 서버의 정보를 검색하거나 저장, 삭제 등에 관련된 제어 기능을 구현하는데 주로 사용된다.

2.4.3. JSP(Java Server Page)

JSP 는 동적인 웹 문서를 포함하는 자바 기술로, MS 사의 ASP(Active Server Page)와 거의 같다. JSP 는 서블릿의 기능을 모두 포함하고 있으며, 자바 빈(Bean)를 사용할 수도 있다. 그러나 웹 어플리케이션에서 JSP 는 서블릿과 같이 사용하여 구현되며, JSP 는 클라이언트의 요청을 받고 결과를 보여주는데 이용된다.

3. 웹 마이그래이션 접근 방법

웹 마이그래이션 하기 위한 단계와 세부 워크플로우와 활동 및 산출물을 정의한다.

3.1. 웹 마이그래이션 단계

본 논문에서 제시하는 웹 마이그래이션 단계는 RUP 를 기반으로 하여 그림 1 과 같은 단계로 이루어진다.

3.2. 각 단계 정의

3.2.1. 기존 시스템 이해

이 단계는 RUP 에서는 제안하지 않은 단계로, 본 논문에서는 기존 C/S 어플리케이션에 대한 이해를 하기 위한 단계이다. 가장 기초가 되는 요구 사항에서 분석, 설계 단계의 산출물, 실제 어플리케이션을 가지고 시스템을 재정의한다.

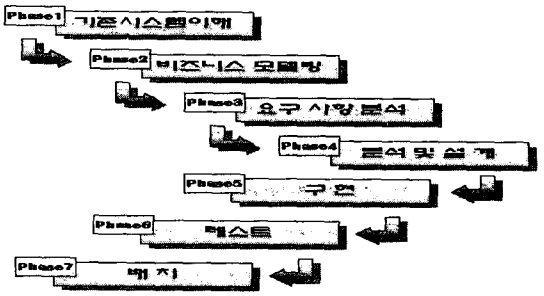

그림 1. RUP 를 적용한 웹 마이그래이션 프로세스

3.2.2. 비즈니스 모델링

기존의 C/S 어플리케이션을 기반으로 한 웹 어플리케이션에 대한 비즈니스 유스케이스 모델(Business Use-Case Model)과 비즈니스 오브젝트 모델(Business Object Model)를 추출하고 정의한다.

3.2.3. 요구사항분석

기존의 C/S 어플리케이션을 재사용하기 위해 요구 사항을 통한 시스템의 범위를 정의한다. 또한 웹으로의 전환 시 사용자의 목적과 필요에 중점을 둔 사용자 인터페이스를 정의한다.

3.2.4. 분석 및 설계

본 논문의 프로세스의 가장 핵심적인 것으로써, 본 단계에서는 분석, 설계 단계의 산출물을 가지고 재사용 클래스를 추출하고, 재사용 클래스를 웹 마이그래이션 분석, 설계 시 적용되는 범위를 정의하고 설계한다. 이 단계에서 4+1 뷰 모델의 아키텍처로 구분해 설계한다.

간단히 재사용 클래스를 추출하는 방법은 우선, 서버측에서 사용되는 클래스는 모든 것이 재사용이 가능하다고 여겨도 될 만큼 재사용성이 높다. 그리고 사용자 인터페이스(클라이언트측)를 포함하는 클래스는 재사용이 거의 어렵다. 이런 경우 분석가나 개발자는 웹 인터페이스를 어떤 기술로 할 것인가에 따라서 재사용성의 변화가 있을 수 있다. 마지막으로 통신 프로토콜에 관련된, 서버와 클라이언트를 연결해주고 데이터를 서로 전달해주는 기능을 포함하는 클래스에 대한 변경이다. 웹 기술에 의존하여 재사용성이 다르다.

3.2.5. 구현

앞 단계의 모든 산출물을 가지고 구현하게 되는데, 구현 시 세 부분으로 나눠 구현한다. 첫째, 서버에서의 비즈니스 로직은 웹 서버를 고려해 개발한다. 이 또한, 개발 언어에 따라 재사용에 큰 변화를 가져온다. 둘째는 웹과 기존 어플리케이션 간의 데이터 전송을 위한 중간적인 클래스가 필요하며, 자바를 기반으로 한 어플리케이션에서는 이 부분이 서블릿이 될 수 있다. 셋째, 클라이언트에게 보여주는 부분인데 여러 웹 기술 중에 하나를 선택해 웹 페이지를 작성한다.[8]

3.2.6. 테스트

기존 어플리케이션의 기능과 웹으로 전환 시 기능이 동일한지 검증하고, 웹 어플리케이션이 사용자 요구 사항에 올바르게 부합되는지를 테스트한다.

3.2.7. 배치

클라이언트가 웹 마이그래이션이 된 산출물을 사용하는데

보장하기 위해 관련된 활동을 말하며, 클라이언트가 웹을 통해 올바르게 접근할 수 있도록 배치한다.

4. 사례 연구

본 논문에서 제시한 웹 마이그래이션 접근 방법을 적용해 기존의 C/S로 구현된 게시판을 웹으로 마이그래이션하는 과정이다.

4.1. 설계 모델에서의 재사용

기존 어플리케이션의 산출물인 클래스 다이어그램을 분석하여 웹 마이그래이션 시 재사용 클래스를 추출하는 과정이다.

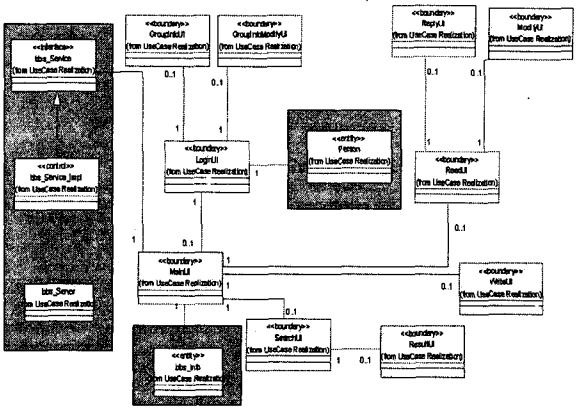

그림 2. 기존 산출물을 통한 재사용 클래스 추출

그림 2의 클래스 다이어그램에서 박스안에 있는 클래스는 모든 것을 재사용이 가능하다. 이 결과로 알 수 있는 것은 스테레오 타입이 <<boundary>>인 경우 완벽한 재사용이 어렵고, 서버측과 스테레오 타입이 <<entity>>인 클래스는 모두 재사용이 가능함을 알 수 있다. 또한 클라이언트가 서버에서 서비스 객체를 얻어오는 MainUI 클래스는 하나의 빈으로 만들거나, 서블릿으로 변경해 객체를 얻어오게 할 수 있다.

4.2. 웹 마이그래이션 설계

재사용 가능한 클래스를 가지고 스퀀스 다이어그램을 다시 그리게 된다. 이런 경우, 우리는 웹 어플리케이션 개발 시 사용할 수 있는 WAE(Web Application Extension)를 사용하여 더 정확한 설계를 할 수 있다.

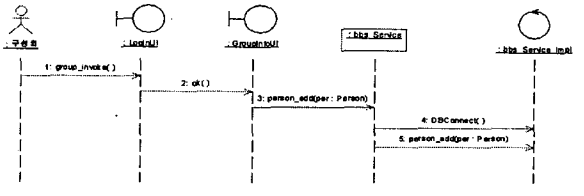

그림 3. 기존 산출물인 [로그인] 시퀀스 다이어그램

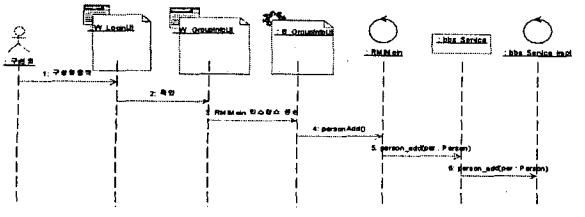

그림 4. WAE를 적용한 [로그인] 시퀀스 다이어그램

그림 3는 C/S 어플리케이션의 시퀀스 다이어그램이며, 그림 4는 웹 마이그래이션 시의 시퀀스 다이어그램이다.

5. 평가

본 논문에서 적용한 설계 모델 재사용은 객체지향으로 분석, 설계된 기존 어플리케이션을 가지고 사례 연구를 하였다. 클래스에 대한 재사용의 결과는 표 1과 같다. 이 과정에서 정확히 분석된 산출물을 가지고 재사용 클래스를 추출하는 과정에서 몇 가지의 규칙을 찾아낼 수 있었다. 첫째는 사용자 인터페이스에 관련된 클래스는 바로 재사용할 수 없다. 둘째, 서버측 클래스는 재사용이 바로 가능하며, 클라이언트 클래스는 변경해야만 한다. 셋째, C/S 어플리케이션과 웹 어플리케이션 간의 통신을 위한 클래스는 최대한 분리해 설계해야 한다.

사례 연구에서 기존 어플리케이션은 계속 사용하면서 웹으로도 병렬적으로 서버를 같이 사용하여 서비스할 수 있었다.

표 1. 웹 마이그래이션 시 재사용에 대한 결과

기존 어플리케이션의 클래스 수	웹 마이그래이션 한 어플리케이션		
	완전 재사용된 클래스	변형된 클래스	새로 생성한 클래스
총 16개	5개	1개(50%이하변경시)	10개(50%이상 변경시)

6. 결론 및 향후 연구

기존 시스템을 웹 마이그래이션 하는 단계는 기존의 C/S로 개발된 어플리케이션을 최대한 재사용하는 것을 도움으로써 개발의 비용과 시간을 절약하고 웹 어플리케이션 개발의 위험(Risk)를 줄인다. 따라서 많은 기술들에 대한 공통적인 프로세스가 정의되어야 하며, 수많은 웹 관련 기술과 기존 어플리케이션 간의 통신을 위한 공통적인 클래스, 즉 기존 시스템을 최대한 재사용할 수 있는 랩퍼(Wrapper)가 필요하다.[2]

7. 참고 문헌

[1] Ellis Horowitz, "*Migrating Software to the World Wide Web*", IEEE software, May/June, 1998, pp.18~21

[2] Frank P. Coyle, "*Legacy Integration-Changing Perspectives*", IEEE Software, March/April, 2000. pp.37~41

[3] Philippe Kruchten, *The Rational Unified Process An Introduction*, Addison-Wesley Longman, Inc., 1998

[4] Philippe Krechten, "*Architectural Blueprints-The 4+1 View Model of Software Architecture*", IEEE Software 12(6), November, 1995, pp.42~50

[5] A Rational Software & Context Intergation white paper, "*Building Web Solutions with the Rational Unified Process: Unifying the Creative Design Process and the Software Engineering Process*", Rational Software Corporation, 1999

[6] Jim Conallen, "*Building web Applications with UML*", Addison-Wesley Longman, Inc., 1999

[7] Jim Conallen, *Modeling Web Application Architecture with UML*, Communication of the ACM, vol. 42, No, 10, October, 1999, pp.63~70

[8] Roger Fournier, *A Methodology for Client/Server and Web Application Development*, Prentice-Hall, Inc., 1999